REMARKS

With this paper, claims 1-27 are pending, of which claims 1, 20, 22, 26, and 27 are independent. Of these, claim 26 is a computer program product claim corresponding to the method recited in claim 1, while claim 27 is a computer program product claim corresponding to the method recited in claim 22. Claims 1, 7, 20-22, 26, and 27 are amended herewith.

The *Office Action* considered claims 1–27. The *Office Action* rejected claims 1–27 under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent Publication No. 20050004942 to Madsen et al., ("*Madsen*.")[1]

As a preliminary matter, Applicants have amended the specification in light of the typographical objections pointed out in the *Office Action*. Applicants respectfully submit, therefore, that these particular objections of record are now moot.

In general, Applicants' invention relates to computer-executable components used in a computer system, where the computer-executable requesting components request and implement functionality from other computer-executable target components. In particular, each component represents one or more sets of computer-executable instructions, which can form part of an overall program or larger executable component, and can also access and/or execute (requesting components) still other executable components (target components). As discussed in Applicants' Background section, at least one of the problems to which Applicants' invention is addressed involves the notion that there can sometimes be a mismatch between what a requesting

[1] Although the prior art status of the cited art is not being challenged at this time, Applicants reserve the right to challenge the prior art status of the cited art at any appropriate time, should it arise. Accordingly, any arguments and amendments made herein should not be construed as acquiescing to any prior art status of the cited art.

application or component needs to execute and what is available in the sysetm. This problem can be exacerbated where requesting components request target components that are out-of-date, or are replaced due to some component upgrades or changes in the computerized system.

Accordingly, Applicants' invention generally involves a number of different mechanisms to ensure that requesting components, when running, can receive, access, and otherwise execute appropriate versions of a given target component upon request. For example, a determination module receives one or more requests from a requesting component for a specific version of a target component. After receiving the request, the determination module then checks a versioning policy associated with the requested target component (and/or versioning policy of another version of the target component). Based on the identified versioning policies for each target component of interest, the determination module can then determine and provide an appropriate version of the given target component to the requesting component, and ensure the requesting component has what it needs. Thus, the determination module does not necessarily know in advance what a requesting component needs, but will alternatively make ongoing determinations, based on versioning policies in the requested target component(s), as needed in response to a request.

Thus, versioning policies can be considered an important aspect of the present invention. In particular, versioning policies can determine not only what version of a target component is provided on request, but also the manner in which various components are upgraded, so as to avoid future component conflicts during execution. As discussed in Applicants' specification, for example, some components, such as library components, cannot be replaced during an upgrade, while other components, such as platform components, can be replaced during an upgrade. Whether a given component is overwritten, or simply placed beside a different version of itself, depends at least in part on the versioning policy associated with that particular component. Applicants have amended the claims to clarify these and other features of the invention.

Accordingly, Applicants' invention as recited in amended independent claim 1 (and corresponding computer program product claim 26) includes, in a computerized system that includes one or more program computer-executable components, including one or more computer-executable requesting components configured to execute one or more computer-executable target components in the computerized system, a method of providing a computer-executable requesting component with access to an appropriate version of a computer-executable target component upon request, comprising the acts of receiving a request from a requesting component for access by the requesting component of a specified version of a computer-executable target component; upon receiving the request from the requesting component, identifying a versioning policy of the specified version of the requested target component; identifying an appropriate version of the target component based on the versioning policy of the specified target component; and providing the requesting component with access to the appropriate version of the target component, wherein the requesting component executes the identified and provided target component.

In addition, Applicants' invention as recited in amended independent claim 20 includes, in a computerized system that includes one or more program computer-executable components, including one or more computer-executable requesting components that can request to access one or more computer-executable target components in the computerized system, a method of providing a computer-executable requesting component with access to an appropriate version of a computer-executable target component, comprising receiving a request from a requesting component for access by the requesting component of a specified version of a computer-executable target component; a step for, upon receiving the request from the requesting component, determining an appropriate version of the requested target component based on a versioning policy corresponding to the requested target component, and allowing access to the appropriate version of the requested target component such that the requesting component accesses the appropriate target component as it has been configured to do so, and such that the requesting component does not fail when requesting access to a component that has been upgraded.

Furthermore, Applicants' invention as recited in amended independent claim 22 (and corresponding computer program product claim 27) includes, in a computerized system that includes one or more program components, including one or more requesting components that can request to access one or more target components in the computerized system, a method of upgrading a computer-executable target component such that a computer-executable requesting component that accesses the computer-executable target component continues to operate effectively after the target component has been upgraded, comprising the acts of identifying that a requesting component is configured to execute a computer-executable target component; identifying a versioning policy in at least an existing version of the target component and a versioning policy in a previously installed version of the target component; and identifying which versions of the target component should remain on the system based on any of the identified versioning policies corresponding to at least the existing version of the target component and the previously installed version of the target component.

By contrast, the *Madsen* reference is directed to a different problem than that solved by Applicants. In particular, the *Madsen* reference teaches mechanisms for ensuring that physical devices are appropriately operable on a general network (*i.e.*, rather than the interoperability of components in a computer system). *E.g.*, ¶ 0052. For example, in the Abstract, the *Madsen* reference teaches a "network management system" "that allows a user to configure multiple devices according to a consistent set of policies." Similarly, the Summary discloses mechanisms "for managing the configuration devices on a network, through subscriptions to a common database of policies." ¶ 0016. As a preliminary matter, therefore, *Madsen* fails to disclose a *computer-executable* requesting component and a *computer-executable* target component, as recited in independent claims 1, 20, 22, 26, and 27.

In any event, the *Madsen* reference teaches that, to ensure device operability on a network, a "component database" stores a set of configuration profiles for each physical device on a network. ¶¶ 0053-0054. For example, the *Madsen* reference teaches that a "Device Learning System" "reads configuration data from a network device" and creates profiles for each device based on available system components. *E.g.*, Abstract, ¶¶ 0068-0070. In particular, the Device Learning System "retriev[es] and analys[es]" "the native configuration running on the device 10 at the time of import." In addition, *Madsen* teaches that the Device Learning System compares the native device configuration (*i.e.*, an "instance tree") with one or more components or policies in the system component database. ¶¶ 0085-0087. The Device Learning System then supplements the native instance tree with matching or candidate system components to create a "complete" instance tree, which is then stored or updated as needed in the component database. ¶ 0093.

Each disclosed instance of this scanning and/or updating of the instance tree occurs in advance of actual use of the physical device on the network, or during a time in which the physical device is in an appropriate "state." *E.g.*, ¶¶ 0251, 0253, 0255. Applicants respectfully submit, therefore, that nothing in the *Madsen* reference teaches that a requesting component requests access of a *specific version* of a target component, and, upon making the request, is then provided access of an appropriate version of a target component, based on a determination of the given target's versioning policy. For example, even construing the physical device that is imported into the network as a "requesting component" in "the system," nothing in the *Madsen* reference teaches, discloses, or otherwise suggests that the physical device ever requests any access of a particular version of a target component. Furthermore, nothing in the *Madsen* reference teaches that, upon receiving a request from the requesting component (physical device), an appropriate version is provided to the requesting component based on an analysis of a specific versioning policy associated with a specified target component.

Even in the alternative perspective, nothing in the *Madsen* reference teaches that the Device Learning System requests execution of any versions of any computer-executable target components. Similarly, nothing in the *Madsen* reference teaches, discloses, or otherwise suggests that a computer executable target component is returned to the Device Learning System, whereupon, the Device Learning system "executes the provided target component." Rather, the *Madsen* reference makes clear that the Device Learning System simply inspects the configuration of the physical device to create an appropriate device profile (or "instance tree") that is appropriately configured for the network. Although it is ultimately the physical device that implements the device profile, nothing in the *Madsen* reference teaches that the physical device ever makes a request, as such, to execute a specific version of a target component in the computer system.

The *Office Action* cites ¶¶ 0053-0055 for the limitation of receiving a request from a requesting component for a specific version of a target component. Applicants respectfully traverse this citation. In particular, Applicants respectfully suggest that these and other passages throughout the *Madsen* reference appear to teach or suggest that the disclosed system simply updates instance trees for a given device as needed *in advance* of any operational needs. *E.g.*, ¶ 0236-0237, 0265 (discussing "periodic auditing.") In some cases, these updates to the device's components are simply pushed to the device when the device is in a "predictable state." *E.g.*, ¶¶ 0251, 0253, 0255. Thus, when the device attempts to operate using a policy that is now updated in the system, the device is automatically referred to the updated policy, rather than the original policy. This purportedly helps "decreas[e] the effort required to run a network using the system." ¶¶ 0059, 0072.

Applicants have therefore amended independent claims 1, 20, 22, 26, and 27 to clarify these various distinctions. In particular, Applicants respectfully submit that the Madsen reference fails to teach, disclose, or otherwise suggest "receiving a request from a requesting component for access by the requesting component of a specified version of a computer-executable target component." Accordingly, claims 1, 20, and 26 (and the corresponding dependent claims) are allowable for at least these reasons. Similarly, Applicants respectfully submit that *Madsen* fails to teach "upon receiving the request from the requesting component, identifying a versioning policy of the specified version of the requested target component," nor "identifying a versioning policy in at least an existing version of the target component and a previously installed version of the target component."

In particular, Applicants respectfully submit that the *Madsen* reference fails to teach a "versioning policy," much less that a "versioning policy" is specifically associated with a particular version of a target component. For example, *Madsen* teaches that each physical device can be associated with one or more "policies," which identify certain preferences and configurations for operability. In particular, *Madsen* teaches that "policy-driven configurations 16 are data structures which represent the *total desired state of a network device* 10 with the system." ¶ 0054. (Emphasis added). In addition, "policy-driven configurations 16 are a set of references or pointers to instances 20 of components 26 stored elsewhere in the component database 28, or policies 34." ¶ 0055. *Madsen* adds that "policies 34 are persistent groups of instances 20 which can be reused across many [policy-driven] configurations 16." *Id.* *Madsen* further adds that:

> Policies 34 are the means by which configurations can be factored into larger-scale units and reused. Policies 34 create a "change once, apply everywhere" semantic to network device configuration, and are the principal mechanism for decreasing the effort required to run a network using the system.

¶ 0059. Policies, as taught by *Madsen*, thus appear to be broadly associated with many devices and/or many different configurations or components.

This, however, is wholly different and distinct from the "versioning policy" disclosed by Applicants. For example, Applicants teach that each component has its own "versioning policy," and that each versioning policy indicates how a specific version of a target component can be used. In particular:

> A "versioning policy", for the purposes of this specification and claims includes any of a given set of properties that can be conveyed from a target component (e.g., 120, 125, 130) to a determining module 100. The versioning policy 131, 132, or 133 specifies whether the corresponding target component 120, 125, 130 can be used instead of a version of the given target component with a lower version number. The versioning policy can include additional information intended to be used by the determining module 100 to decide whether the target component can be used in a given configuration. Thus, the versioning policy 132 may specify that target component 125 (version 1.2) can be used when version 1.1 is requested.

Applicants' specification, ¶ 0034. Rather than representing "persistent groups of instances," as disclosed in *Madsen*, each "versioning policy," as taught by Applicants, is associated with a specific target component.

The *Office Action* cites ¶¶ 0028, 0061, and 0065 for the limitation of identifying versioning policies, such as taught in independent claim 22. Applicants, however, respectfully traverse this citation. In particular, Applicants respectfully submit that these passages simply teach that each policy, which can be applied broadly to a set of devices or components, can include version information as generally applied to several components and/or several devices associated with a device configuration profile. The version information disclosed by *Madsen* appears to be geared primarily for allowing users to potentially roll back one policy to an earlier version of the policy, or to push a newer version of a policy to a set of one or more devices. *E.g.*, ¶¶ 0065, 0236-0237.

Accordingly, Applicants can find nothing in the *Madsen* reference that teaches or suggests identifying that a computer-executable requesting component is configured to execute a particular target component, and then identifying two different versioning policies in correspondingly two different versions of the same target component to identify which versions of the target component should remain on the system. Applicants respectfully submit, therefore, that amended independent claims 22 and 26 (and the corresponding dependent claims) are also allowable for at least these reasons.

As a final matter, nothing in the *Madsen* reference teaches that the requesting component and the target component are part of the same computerized system. Rather, *Madsen* teaches throughout the disclosure that the Device Learning System is configured to determine and apply policy configurations for multiple different devices in a network. As previously mentioned, Applicants respectfully submit that this problem identified by *Madsen* is entirely distinct from that being solved by Applicants' invention. To clarify this point, Applicants have further amended the preambles of each of independent claims 1, 20, 22, 26, and 27 to clarify that the requesting component and the target component are in the same computer system.

In view of the foregoing, Applicants respectfully submit that the other rejections to the claims are now moot and do not, therefore, need to be addressed individually at this time. It will be appreciated, however, that this should not be construed as Applicants acquiescing to any of the purported teachings or assertions made in the last action regarding the cited art or the pending application, including any official notice. Instead, Applicants reserve the right to challenge any of the purported teachings or assertions made in the last action at any appropriate time in the future, should the need arise. Furthermore, to the extent that the Examiner has relied on any Official Notice, explicitly or implicitly, Applicants specifically request that the Examiner provide references supporting the teachings officially noticed, as well as the required motivation or suggestion to combine the relied upon notice with the other art of record.

In the event that the Examiner finds remaining impediment to a prompt allowance of this application that may be clarified through a telephone interview, the Examiner is requested to contact the undersigned attorney.

Dated this 24th day of April, 2007.

Respectfully submitted,

/Michael J. Frodsham/

RICK D. NYDEGGER
Registration No. 28,651
MICHAEL J. FRODSHAM
Registration No. 48,699

Attorneys for Applicant
Customer No. 047973

MJF:kbl
W:\13768\493\KJB0000009768V001.DOC